Name: Santos, Tristan Neal U.
Course and Section: CPE32S9
Date of Submission: 02-07-2024
Instructor: Engr. Roman Richard

In this assignment, you are task to build a multilayer perceptron model. The following are the requirements:

- Choose any dataset
- Explain the problem you are trying to solve
- Create your own model
- Evaluate the accuracy of your model

```
#Import and Choose a Dataset
import numpy as np
import pandas as pd

from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split

from sklearn.metrics import classification_report, confusion_matrix

from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

Double-click (or enter) to edit

```
#Display the Dataset to showcase the output and set the head to 10 so the result would reach from 0-9
winedf = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Emtech2/csv/WineQT.csv')
winedf.head(10)
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	Id
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	0
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5	1
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5	2
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6	3
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5	4
5	7.4	0.66	0.00	1.8	0.075	13.0	40.0	0.9978	3.51	0.56	9.4	5	5
6	7.9	0.60	0.06	1.6	0.069	15.0	59.0	0.9964	3.30	0.46	9.4	5	6
7	7.3	0.65	0.00	1.2	0.065	15.0	21.0	0.9946	3.39	0.47	10.0	7	7
8	7.8	0.58	0.02	2.0	0.073	9.0	18.0	0.9968	3.36	0.57	9.5	7	8
9	6.7	0.58	0.08	1.8	0.097	15.0	65.0	0.9959	3.28	0.54	9.2	5	10

Next steps: View recommended plots

```
winedf.describe().transpose()
```

	count	mean	std	min	25%	50%	75%	max
fixed acidity	1143.0	8.311111	1.747595	4.60000	7.10000	7.90000	9.100000	15.90000
volatile acidity	1143.0	0.531339	0.179633	0.12000	0.39250	0.52000	0.640000	1.58000
citric acid	1143.0	0.268364	0.196686	0.00000	0.09000	0.25000	0.420000	1.00000
residual sugar	1143.0	2.532152	1.355917	0.90000	1.90000	2.20000	2.600000	15.50000
chlorides	1143.0	0.086933	0.047267	0.01200	0.07000	0.07900	0.090000	0.61100
free sulfur dioxide	1143.0	15.615486	10.250486	1.00000	7.00000	13.00000	21.000000	68.00000
total sulfur dioxide	1143.0	45.914698	32.782130	6.00000	21.00000	37.00000	61.000000	289.00000
density	1143.0	0.996730	0.001925	0.99007	0.99557	0.99668	0.997845	1.00369
pH	1143.0	3.311015	0.156664	2.74000	3.20500	3.31000	3.400000	4.01000
sulphates	1143.0	0.657708	0.170399	0.33000	0.55000	0.62000	0.730000	2.00000
alcohol	1143.0	10.442111	1.082196	8.40000	9.50000	10.20000	11.100000	14.90000
quality	1143.0	5.657043	0.805824	3.00000	5.00000	6.00000	6.000000	8.00000
Id	1143.0	804.969379	463.997116	0.00000	411.00000	794.00000	1209.500000	1597.00000

```
print(winedf.head())
print(winedf['quality'].unique())
```

```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
3                 17.0                  60.0   0.9980  3.16       0.58
4                 11.0                  34.0   0.9978  3.51       0.56

   alcohol  quality  Id
0      9.4        5   0
1      9.8        5   1
2      9.8        5   2
3      9.8        6   3
4      9.4        5   4
[5 6 7 4 8 3]
```

```
winedf.describe(include = 'all')
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	Id
count	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000
mean	8.311111	0.531339	0.268364	2.532152	0.086933	15.615486	45.914698	0.996730	3.311015	0.657708	10.442111	5.657043	804.969379
std	1.747595	0.179633	0.196686	1.355917	0.047267	10.250486	32.782130	0.001925	0.156664	0.170399	1.082196	0.805824	463.997116
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000	3.000000	0.000000
25%	7.100000	0.392500	0.090000	1.900000	0.070000	7.000000	21.000000	0.995570	3.205000	0.550000	9.500000	5.000000	411.000000
50%	7.900000	0.520000	0.250000	2.200000	0.079000	13.000000	37.000000	0.996680	3.310000	0.620000	10.200000	6.000000	794.000000
75%	9.100000	0.640000	0.420000	2.600000	0.090000	21.000000	61.000000	0.997845	3.400000	0.730000	11.100000	6.000000	1209.500000
max	15.900000	1.580000	1.000000	15.500000	0.611000	68.000000	289.000000	1.003690	4.010000	2.000000	14.900000	8.000000	1597.000000

```
x = winedf.drop(['Id', 'quality'], axis=1)
y = winedf['quality']

print(x.head())
print(y.head())
```

```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                 11.0                  34.0   0.9978  3.51       0.56
1                 25.0                  67.0   0.9968  3.20       0.68
2                 15.0                  54.0   0.9970  3.26       0.65
3                 17.0                  60.0   0.9980  3.16       0.58
4                 11.0                  34.0   0.9978  3.51       0.56

   alcohol
0      9.4
1      9.8
2      9.8
3      9.8
4      9.4
0    5
1    5
2    5
3    6
4    5
Name: quality, dtype: int64
```

```
#Split Database/Dataset to Train and Test
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
print(x_train.shape)
print(y_train.shape)
```

```
(800, 11)
(800,)
```

```
#Train the Model
mlp = MLPClassifier(hidden_layer_sizes=(3,2), max_iter=1000, activation='relu')



mlp.fit(x_train,y_train)
```

```
MLPClassifier
MLPClassifier(hidden_layer_sizes=(3, 2), max_iter=1000)
```

```
#Testing
pred = mlp.predict(x_test)
pred

#The array signifies the predicted quality scores from each sample in the test set.
#The score array is the result from the wine quality class
```

```
array([5, 6, 5, 6, 5, 6, 6, 6, 5, 6, 5, 6, 6, 6, 6, 5, 6, 6, 6, 6, 6,
       6, 5, 6, 6, 6, 5, 6, 5, 6, 5, 6, 5, 5, 6, 5, 6, 5, 5, 6, 6, 6, 5,
       5, 5, 6, 5, 6, 6, 6, 5, 5, 6, 6, 6, 5, 6, 6, 6, 5, 6, 6, 6, 6, 6,
       5, 6, 5, 5, 6, 5, 5, 6, 6, 5, 5, 6, 6, 6, 5, 6, 5, 6, 6, 6, 6, 6,
       6, 6, 6, 6, 5, 6, 6, 6, 6, 6, 5, 6, 5, 5, 5, 5, 6, 6, 5, 6, 6, 5,
       6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 5, 5, 6, 5, 6, 6, 6, 6, 5, 6, 5, 6,
       6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6, 6, 6, 5, 6, 6, 5, 6, 5, 6, 6,
       6, 6, 6, 6, 5, 6, 6, 5, 6, 6, 5, 6, 6, 5, 6, 6, 6, 6, 5, 6, 6, 5,
       6, 5, 5, 5, 6, 6, 6, 6, 5, 5, 6, 6, 6, 6, 5, 5, 6, 6, 6, 6, 6, 5,
       6, 5, 6, 6, 6, 6, 6, 5, 6, 5, 6, 6, 5, 6, 6, 6, 5, 6, 6, 6, 5, 6,
       5, 6, 6, 6, 6, 5, 6, 6, 6, 5, 6, 5, 6, 6, 6, 6, 6, 5, 6, 5, 5, 6,
       6, 6, 6, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 5, 5, 6, 5, 6, 5, 5,
       6, 6, 6, 6, 6, 5, 6, 6, 5, 5, 6, 6, 6, 6, 6, 5, 5, 6, 6, 6, 5, 6,
       5, 6, 5, 5, 6, 6, 6, 5, 5, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6, 5, 5, 6,
       6, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 5, 6, 5, 5, 6, 6, 6, 6, 5, 6, 5, 5,
       6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6])
```

```
#Model Evaluation
#The arrays display is based from its number quality on the dataset used with different number of samples
#As it showcase 0 with other different numbers.
confusion_matrix(y_test,pred)
```

```
array([[  0,   0,   0,   3,   0,   0],
       [  0,   0,   3,   9,   0,   0],
       [  0,   0,  61,  79,   0,   0],
       [  0,   0,  38, 103,   0,   0],
       [  0,   0,   6,  38,   0,   0],
       [  0,   0,   0,   3,   0,   0]])
```

```
#This output will showcase the precision, recall, f1-score, support and its accuracy, macro avg, and weighted avg.
#From its accuracy, it has a 0.48% precise results wheter this is correct or not while the macro avg is at 017 and lastly weighted avg is 0.43 with the support of 343.
print(classification_report(y_test, pred, zero_division=1))
```

```
              precision    recall  f1-score   support

           3       1.00      0.00      0.00         3
           4       1.00      0.00      0.00        12
           5       0.56      0.44      0.49       140
           6       0.44      0.73      0.55       141
```

```
                7       1.00      0.00      0.00        44
                8       1.00      0.00      0.00         3

         accuracy                           0.48       343
        macro avg       0.83      0.19      0.17       343
     weighted avg       0.59      0.48      0.43       343
```

Conclusion:

From the experienced of this activity, I have a good understanding about how we build and Apply Multilayer Perceptron on the csv dataset. This lesson is similar from the previous lessons since it also involves about using datasets with a different result depending on the code, but this one we also used from the previous lesson with the train test split but the difference here is that we used the classification report and mlp commands that also feature the results of its accuracy of the dataset.